WEST Search History

Hide Items | Restore | Clear | Cancel

DATE: Friday, March 17, 2006

Hide?	Set Name	Query	Hit Count
		DB=PGPB,USPT,USOC,EPAB,JPAB,DWPI,TDBD; PLUR=YES; OP=ADJ	
☐	L14	L11 and (load balancing)	8
☐	L13	L11 and (distributed server)	0
☐	L12	L11 and (distributed content)	0
☐	L11	L10 and @AD<20001030	81
☐	L10	queue near3 (service or user or client) near3 request near3 (notify or notification or delay or wait or waiting or delaying)	139
☐	L9	L8 and @AD<20001030	125
☐	L8	queue near5 (service or user or client) near3 request near5 (notify or notification or delay or wait or waiting or delaying)	216
☐	L7	L5 NOT l6	8
☐	L6	L5 and (queue or queuing)	12
☐	L5	L4 and (load balancing)	20
☐	L4	L3 and (replica or replication)	142
☐	L3	L2 and @AD<20001030	2503
☐	L2	(wait or waiting or delay or delaying) adj (message or notification or (electronic mail) or e-mail)	4914
☐	L1	(wait or waiting or delay or delaying) near3 (message or notification or (electronic mail) or e-mail)	20456

END OF SEARCH HISTORY

WEST Search History

Hide Items | Restore | Clear | Cancel

DATE: Friday, March 17, 2006

Hide?	Set Name	Query	Hit Count
	DB=PGPB,USPT,USOC,EPAB,JPAB,DWPI,TDBD; PLUR=YES; OP=ADJ		
☐	L14	L13 and l10	3
☐	L13	content near5 (deliveray or management or managing or delivering)	27011
☐	L12	L10 and l4	0
☐	L11	L10 and l6	1
☐	L10	L9 and @AD<20001030	88
☐	L9	(load or capacity or bandwidth) near5 (threthold or predetermined) near8 (queue or queuing)	190
☐	L8	L7 and @AD<20001030	5
☐	L7	((queue or queuing) near3 (service request)) and (delay adj2 (notification or message))	9
☐	L6	(delay adj2 (notification or message))	2522
☐	L5	((queue or queuing) near3 (service request)) same (delay adj2 (notification or message))	1
☐	L4	L3 and @AD<20001030	16
☐	L3	(content near3 (managment or management or managing)) near8 (load near3 (information or data or balancing))	49
	DB=PGPB,USPT; PLUR=YES; OP=ADJ		
☐	L2	(content near3 (deliver or delivery)) near8 (load near3 (information or data or balancing)) near8 (plurality adj2 server)	1
	DB=USPT; PLUR=YES; OP=ADJ		
☐	L1	(content near3 (deliver or delivery)) near8 (load near3 (information or data or balancing)) near8 (plurality adj2 server)	0

END OF SEARCH HISTORY

Previous Doc Next Doc Go to Doc#
First Hit Fwd Refs

Generate Collection

L4: Entry 2 of 16 File: USPT Feb 17, 2004

DOCUMENT-IDENTIFIER: US 6694358 B1
TITLE: Performance computer network method

Application Filing Date (1):
20000823

Detailed Description Text (19):
In the present embodiment, POP servers 10-30 and customer web server 40 and 45 work together to provide static content and dynamic content to a user, such as user 150 coupled to ISP 60. The POP network illustrated in FIG. 1 provides content hosting for static content, such as images, video, software, etc. The POP network also provides for load balancing and traffic management for services, and dynamic content, that cannot be stored. Further detail regarding the process will be described below.

Previous Doc Next Doc Go to Doc#

Previous Doc Next Doc Go to Doc#
First Hit Fwd Refs

Generate Collection

L4: Entry 3 of 16 File: USPT Apr 22, 2003

DOCUMENT-IDENTIFIER: US 6553413 B1
** See image for Certificate of Correction **
TITLE: Content delivery network using edge-of-network servers for providing content delivery to a set of participating content providers

Application Filing Date (1):
20000628

Detailed Description Text (62):
Many existing solutions require active management of content distribution, content replication and load balancing between different servers. In particular, decisions about where content will be hosted must be made manually, and the process of replicating data is handled in a centralized push fashion. On the contrary, the invention features passive management. Replication is done in a demand-based pull fashion so that content preferably is only sent to where it is truly needed. Moreover, the process preferably is fully automated; the ISP does not have to worry about how and where content is replicated and/or the content provider.

Previous Doc Next Doc Go to Doc#

Previous Doc Next Doc Go to Doc#
First Hit Fwd Refs

Generate Collection

L8: Entry 4 of 5 File: USPT Jun 26, 2001

DOCUMENT-IDENTIFIER: US 6252614 B1
TITLE: Software architecture which maintains system performance while pipelining data to an MFP and uses shared DLL

Application Filing Date (1):
19980720

Detailed Description Text (45):
In the second phase, the facsimile subsystem 421 sends a request to the DSRU 315 for a filled face buffer. The facsimile send application 411 has not yet filled the empty face buffer and in this example there are no other face buffers which the facsimile send application has filled. Therefore, the DSRU 315 replies to the facsimile subsystem 421 with a delay message.

Detailed Description Text (46):
In the third phase, the PDL interpreter 311 requests an empty face buffer from the DSRU 315. Since the facsimile send application 411 received the last empty face buffer, the DSRU 315 issues a delay message to the PDL interpreter 311.

CLAIMS:

16. The method of apportioning system resources of a host to a first image data processing process in the host by a data storage and retrieval unit in the host as set forth in claim 12, wherein the delay comprises the data storage and retrieval unit placing the request in a queue, the data storage and retrieval unit servicing one or more other requests, and then the data storage and retrieval unit returning to service requests in the queue.

Previous Doc Next Doc Go to Doc#

Previous Doc Next Doc Go to Doc#
First Hit Fwd Refs

Generate Collection

L14: Entry 3 of 8 File: USPT Mar 27, 2001

DOCUMENT-IDENTIFIER: US 6209018 B1
** See image for Certificate of Correction **
TITLE: Service framework for a distributed object network system

Application Filing Date (1):
19971113

Brief Summary Text (21):
In some embodiments, the service framework also includes a load balancing manager for balancing workloads among workers in a worker pool of a service. Also, the service framework may include a service locator for balancing workloads among clones of a service. Further, the service framework may include a service locator proxy for balancing workloads among clones of service locators that provide handles (e.g., object references) to a service.

Drawing Description Text (25):
FIG. 24 provides a load balancing manager (LBM) interface in accordance with some embodiments of the present invention.

Detailed Description Text (33):
In stage 108, the allocateWorker operation returns an appropriate worker as determined by the service object's load balancing manager (LBM) based on runtime workload statistics of each worker as discussed further below with respect to FIG. 25. In stage 110, the service proxy uses the worker handle to execute the SQL request on the worker, and the worker returns the output from the execution to the service proxy of the client. In stage 112, the service proxy calls releaseWorker on the service object to release the reservation on the worker.

Detailed Description Text (40):
FIG. 7 shows a service object 160 of the server 88 of FIG. 3 in accordance with another embodiment of the present invention. In particular, in response to an allocateWorker call by the service proxy on the service object 160, the service object 160 allocates a worker from its worker pool, workers 162 and 164, to the service proxy so that the service proxy can issue work requests. The service object 160 may use an LBM (load balancing manager), as discussed further below with respect to FIG. 25, to select a worker for the service proxy, then the service object 160 calls newClient on the selected worker 162 to notify the worker of the reservation. Also, if the service proxy wants to release the reserved worker 162, then the service proxy calls releaseWorker on the service object 160, and the service object 160 calls clientReleased to notify the worker 162. Each service object (in some embodiments, there is only one service object per service) controls its own pool of workers. Thus, the service object 160 controls the pool of workers including workers 162 and 164. In some embodiments, worker allocation is implemented to support a variety of modes of access to workers, provide fast response time, and balance the workload from client requests across all workers in the worker pool.

Detailed Description Text (92):
FIG. 24 provides a load balancing manager (LBM) interface in accordance with some

embodiments of the present invention. The LBM interface may be written in IDL as provided in FIG. 24. The LBM is an entity of the service object. In particular, the service object instantiates an LBM to manage the pool of workers. The LBM may be a plug-in object that can be customized or entirely replaced.

Detailed Description Text (101):
FIGS. 26A-26B are a flow diagram illustrating the call sequence operation in accordance with some embodiments of the present invention. In stage 400, assuming a client proxy does not already have a reserved worker for a service request, the service proxy obtains a service handle from the service locator. In stage 402, the service proxy issues the allocateWorker call with a set of reservation properties and an uninitialized reservation context. In stage 404, the service object forwards the request to the LBM 360. In stage 406, the LBM determines if any clients are waiting. If clients are waiting, then the LBM enqueues the request in the client wait queue in stage 408. Otherwise, the LBM proceeds to stage 410.

CLAIMS:

14. A computer implemented method for providing a service framework in a distributed object network system, the method comprising:

executing a first set of computer instructions in a first computer, the first set of computer instructions providing access to a service by allocating a worker in a worker pool for the service in response to a service request, wherein the first set of computer instructions provides workload balancing among a plurality of workers in the worker pool for the service, and the service provides access to a limited resource that resides on the first computer;

executing a second set of computer instructions in a second computer, the second set of computer instructions encapsulating the operation of performing a service request for the second computer, wherein the second computer connects to the first computer over a network and further wherein said executing the first set of computer instructions further comprises:

allocating reservations among at least two workers in the worker pool for the service based on worker statistics; and

providing a load balancing manager that comprises at least two queues that each comprise workers that have various properties, wherein each queue comprises a sub-queue that comprises workers that have various priorities.

15. A computer-readable medium comprising software for a service framework for a distributed object network system, the service framework software comprising:

a set of objects, the set of objects providing access to a service, the service providing access to a limited resource residing on a first computer,

wherein the set of objects further comprises:

a plurality of workers in a worker pool for the service;

a load balancing manager that balances workloads among the plurality of workers in the worker pool for the service; and

a service proxy object, the service proxy object encapsulating, for a second computer, the operation of a service request to the set of objects, wherein the second computer connects to the first computer over a network.

16. A computer-readable medium comprising software for a service framework for a distributed object network system, the service framework software comprising:

a set of objects, the set of objects providing access to a service, the service providing access to a limited resource residing on a first computer,

wherein the set of objects further comprises:

a plurality of workers in a worker pool for the service;

a load balancing manager that balances workloads among the plurality of workers in the worker pool for the service; and

a service proxy locator object, the service proxy locator object providing workload balancing among instances of a service locator, wherein the service locator provides an object reference to the set of objects and balances workloads among instances of the requested service.

Previous Doc Next Doc Go to Doc#

Previous Doc Next Doc Go to Doc#
First Hit Fwd Refs

Generate Collection

L18: Entry 2 of 9 File: USPT Feb 24, 2004

DOCUMENT-IDENTIFIER: US 6697475 B1
TITLE: System and method for implementing an end office switch with enhanced functionality using an operating system independent distributed self-contained dynamic logic system

Application Filing Date (1):
20000424

Drawing Description Paragraph Table (2):
TABLE OF CONTENTS I. System Hardware 12 II. System Logic 16 A. Logic Inputs 18 1. Data 19 a. Station Data 19 2. Events 26 a. Edge Switch Point Events 27 b. Transit Network Events 28 c. Time-Out Queue Events 29 B. Logic Outputs-Primitives 29 1. Edge Switch Point Primitives 29 2. Transit Network Primitives 30 3. Database Primitives 32 4. Enhanced Service Primitives 35 5. Time-Out Queue Primitives 36 C. Features 36 1. Three Digit Numbers-N11 36 2. Standard Calling 40 3. Three Digit Features-*AX 45 4. Optional Seven-Digit Equal Access Carrier 45 5. Valid PSTN Address 45 III. Distribution and Activation of System Logic 46 IV. Call Processing Flow 48 A. Outbound Calls 50 B. Inbound Calls 60 V. Anticipated Variations and Modifications 63

Drawing Description Paragraph Table (3):
TABLE 1 ACRONYMS AND DEFINITIONS AIN Advanced Intelligent Network. ATM Asynchronous Transfer Mode. A cell relay transmission scheme used for Broadband Integrated Services Digital Network (B-ISDN) applications. CCS Common Channel Signaling. CRV Call Reference Value is a network end-point address. DLS Dynamic Logic System is the machine independent logic and software application described in this specification. EO End Office. Often referred to as a class 5 switch or central office. The EO is the interface device located at the service providers premises that allows a subscriber to access the PSTN. ESP Edge Switching Point is a programmable switch used to convert the call processing events from the end-user into a format suitable for other network components such as the transit network. ES Enhanced Services provide primarily voice processing capabilities such as voice recording and playback, text to speech, and speech recognition. EVENT A change in signaling (seize, off-hook, on-hook, flash) or a user generated datum (touch-tone). Events are reported when detected. GR-303 Criteria for Next Generation Integrated Digital Loop Carrier (NG-IDLC) systems for concentration and digital transport to and from a RDT located near a subscriber and an IDT located at a Local Digital Switch (LDS) which provides a full range of narrowband and wideband telecommunications services. H.323 A ITU recommendation which describes systems that provide multimedia communications services over Packet Based Networks (PBN). IDT Integrated Digital Terminal (IDT)-is the logical resource of an Local Digital Switch (LDS) that is associated with a single RDT. IN Intelligent Network MGCP Media Gateway Control Protocol is an interface protocol used for controlling VoIP Gateways from external call control elements. PSTN Public Switched Telephone Network RDT Remote Digital Terminal (RDT)-is a GR-303 intelligent network element that provides an interface between the customer lines and the DS1 facilities. The RDT is physically remote from the central office and provides a concentration and digital multiplexing function which reduces costs and improves network reliability. SCP Service Control Point. A transaction processor-based system designed to provide

various network and subscriber database services. SMP Service Management Point. A device that centralizes all functions needed to manage the network and IN services, as well as to customize services, configurations, alarms, generate statistics, and establish access rights. SS7 Signaling System Number 7 STP Signaling Transfer Point. A signaling point with the function of transferring signaling messages from one signaling link to another. TOQ Time-Out Queue TN Transit Network is the part of the PSTN that routes a call from the originating EO to the destination EO.

Detailed Description Text (12):
Having described that portion of a PSTN 8 illustrated in FIG. 1, reference is now directed to FIG. 2. FIG. 2 further illustrates the DLS 14 introduced in FIG. 1. A End Office (EO) system may comprise an Edge Switching Point (ESP) 20, a Service Management Point (SMP) 28, and a Service Control Point (SCP) 32. The SMP 28 may be configured to dynamically distribute logic 35 programmed to implement class 5 functionality through the ESP 20 at the digital matrix switch 2 (see FIG. 1). As further illustrated in FIG. 2, the SMP 28 may be configured to dynamically update or modify logic 35. Having been provided with a plurality of functional programs, the logic 35 may interface with the ESP 20, enhanced services (ES) 26, the SCP 32, a Time-Out Queue (TOQ) 23, and the Transit Network (TN) 30 to connect and service customers integrated with the local loop equipment of the PSTN 8 (not shown). The logic 35 as illustrated in FIG. 2, may be programmed to receive ESP events 21, TOQ events 22, and Transit Network (TN) events 29. Logic 35 may be further programmed to both send and receive ESP primitives 24, ES primitives 27, SCP primitives 33, TOQ primitives 25, and TN primitives 31.

Detailed Description Text (36):
The programmable logic 35 disclosed herein is used to process events. Events originate from several locations in the system of the present invention. More specifically, events originate from an edge station device, the class 4 routing or transit network 30, the integrated time-out queue 23 from within logic 35, and from ES processing ports.

Detailed Description Text (37):
Event processing is a mechanism for receiving an indication that a particular state change has occurred, followed by a stream of logic which may result in waiting for the next event. The logic stream processes the event, sets time-outs, and establishes an application state. As application states are determined, and transitions between application'states occur, the logic 35 will queue transmit primitives to both the edge station device and the class-4 routing or transit network.

Detailed Description Text (60):
c. TIME-OUT QUEUE EVENTS

Detailed Description Text (69):
This primitive may queue the attachment of a DTMF or touch-tone digit detector to the port.

Detailed Description Text (84):
TNE_queue_outdial(reference)

Detailed Description Text (158):
5. TIME-OUT QUEUE PRIMITIVES

Detailed Description Text (289):
Having described that portion of inbound call processing illustrated in FIG. 29, reference is now directed to FIG. 30. In this regard, FIG. 30 is a flowchart illustrating inbound call processing upon a seize event. Seize event processing starts at step 498 as previously described with FIG. 28. Having entered this portion of the flowchart, the logic 35 performs a

station_forward_unconditional_read validity check in step 499. If step 499 is successful, the logic 35 executes a TNE_requeue_outdial in step 501 to forward the call. After forwarding the call, the logic 35 executes a wait for next event in step 159. If step 499 is unsuccessful, the logic 35 performs a station idle check in step 503. If step 503 is successful, the logic 35 executes a TNE_queue_outdial in step 505 to complete the call. Having executed step 505, the logic 35, then executes a TOQ_insert in step 507 to set an answer timer. After setting the answer timer, the logic 35 executes a wait for next event in step 159. If step 503 is unsuccessful, the logic 35 checks the station features in step 509 to determine if the station is configured for call waiting. If step 509 is successful, the logic 35 executes an ESP_switch_connect_tone in step 511 to notify the user that a call is waiting. Next, the logic 35 executes a TOQ_insert in step 513 to set an answer timer. After setting the answer timer, the logic 35 executes a wait for next event in step 159. If step 509 is unsuccessful, the logic 35 performs a station_forward_busy_read check in step 515. If step 515 is successful, the logic 35 executes a TNE_requeue_outdial in step 517 to forward the call. The logic 35 then executes a wait for next event in step 159. If step 515 is unsuccessful, the logic 35 executes a TNE_caller_busy in step 519 to provide busy treatment to the inbound caller. After executing step 519, the logic 35 then executes a wait for next event in step 159.

Previous Doc Next Doc Go to Doc#